DATA MINING 1

Classification

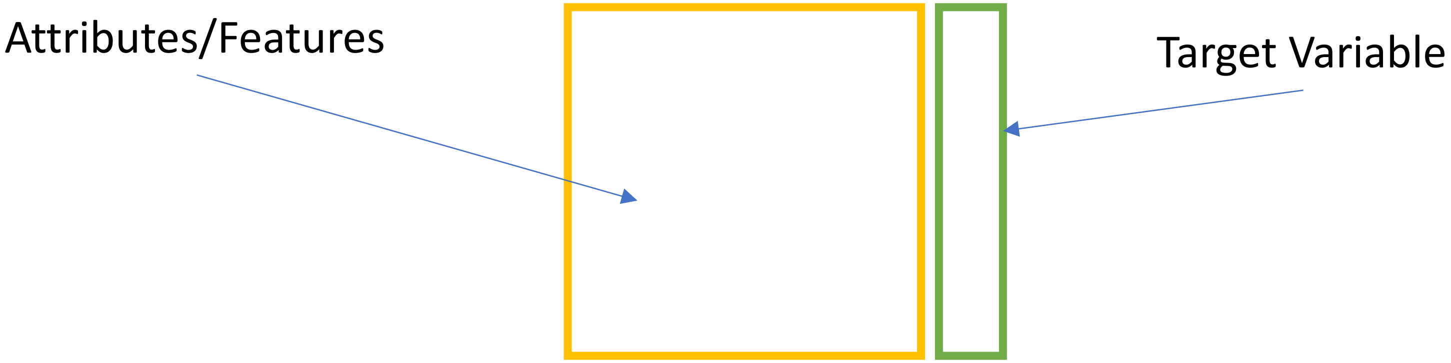
Dino Pedreschi, Riccardo Guidotti

Revisited slides from Lecture Notes for Chapter 3 “Introduction to Data Mining”, 2nd Edition by Tan, Steinbach, Karpatne, Kumar

UNIVERSITÀ DI PISA

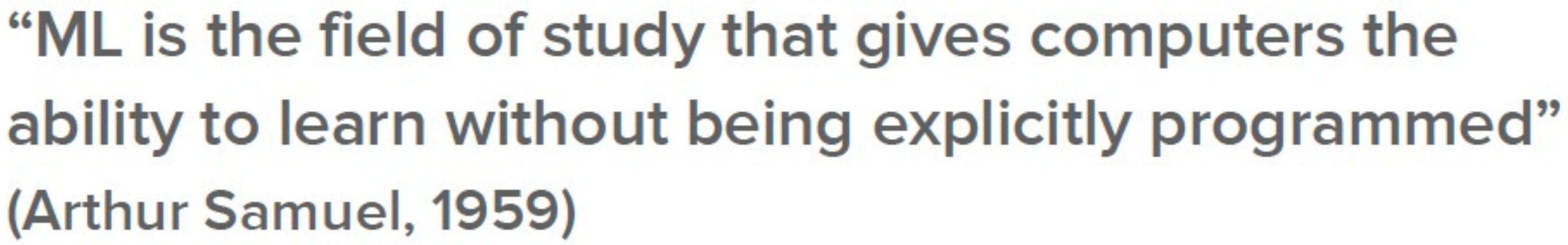
Supervised Learning

- Cluster analysis and association rules are not concerned with a specific target attribute.
- Supervised learning refers to problems where the value of a target attribute should be predicted based on the values of other attributes.
- Problems with a categorical target attribute are called classification, problems with a numerical target attribute are called regression.



What is Machine Learning?

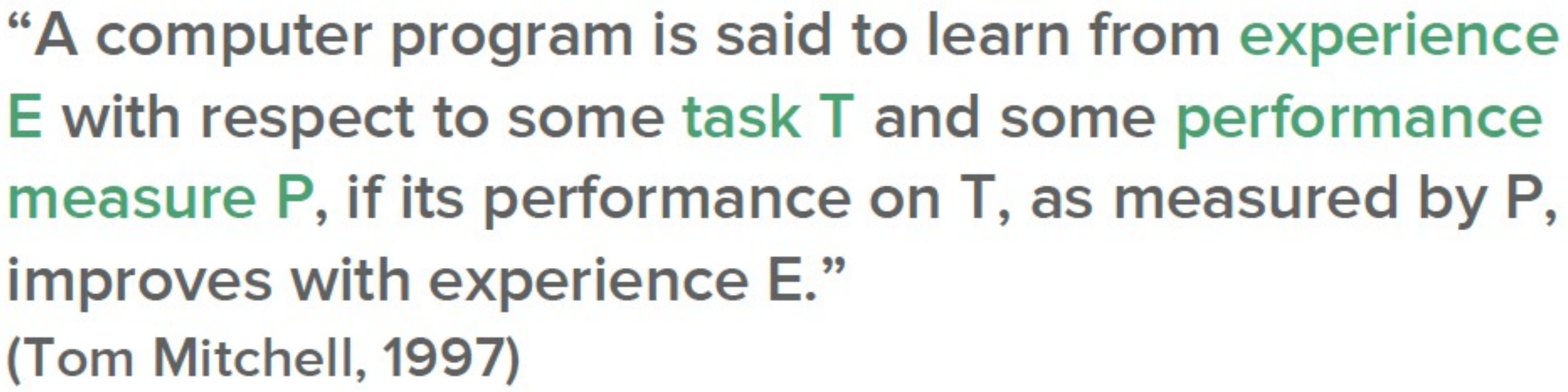
- Machine Learning (ML) is the science (and art) of programming computers that can learn from data.



"ML is the field of study that gives computers the ability to learn without being explicitly programmed"
(Arthur Samuel, 1959)

What is Machine Learning?

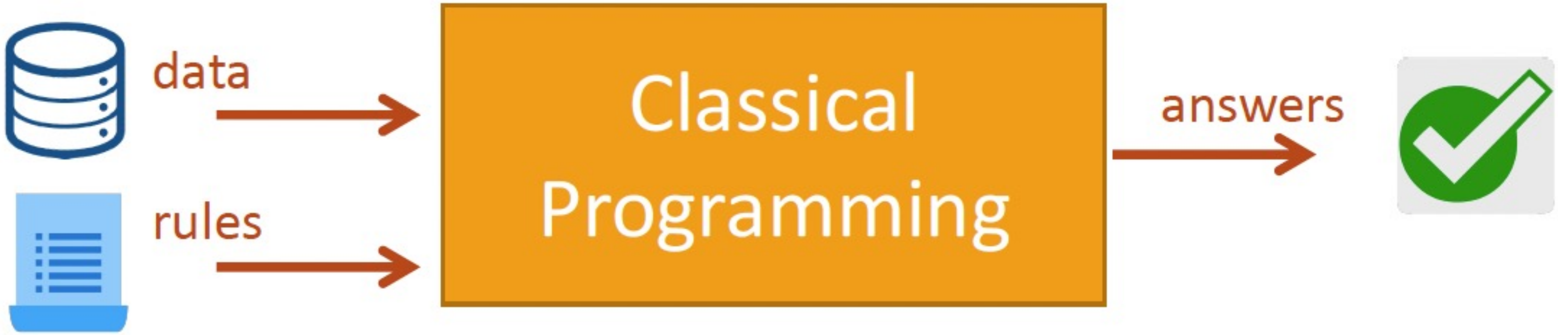
- Machine Learning (ML) is the science (and art) of programming computers that can learn from data.



"A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E."
(Tom Mitchell, 1997)

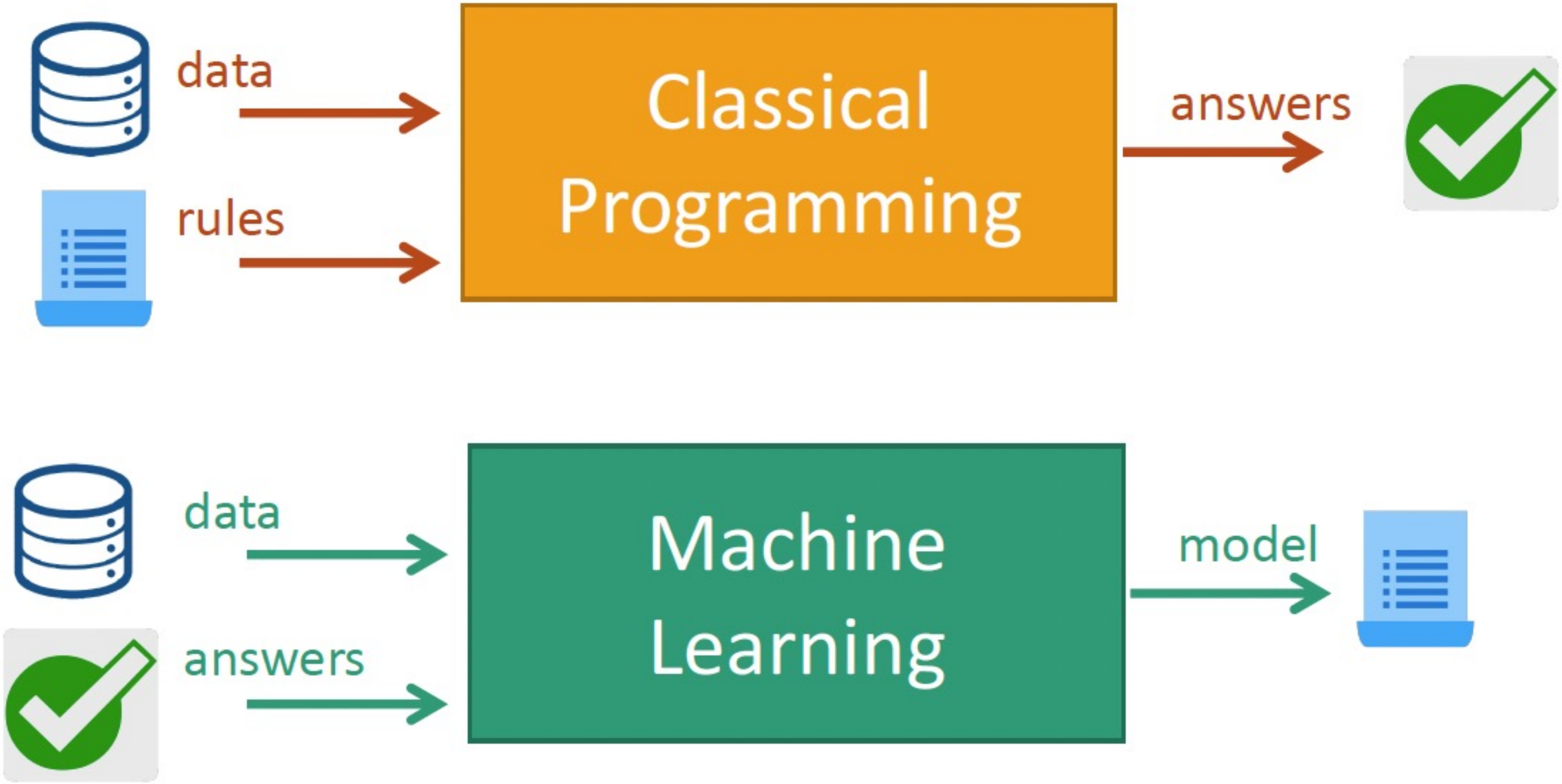
Classical Programming vs Machine Learning

- A ML system is trained rather than programmed



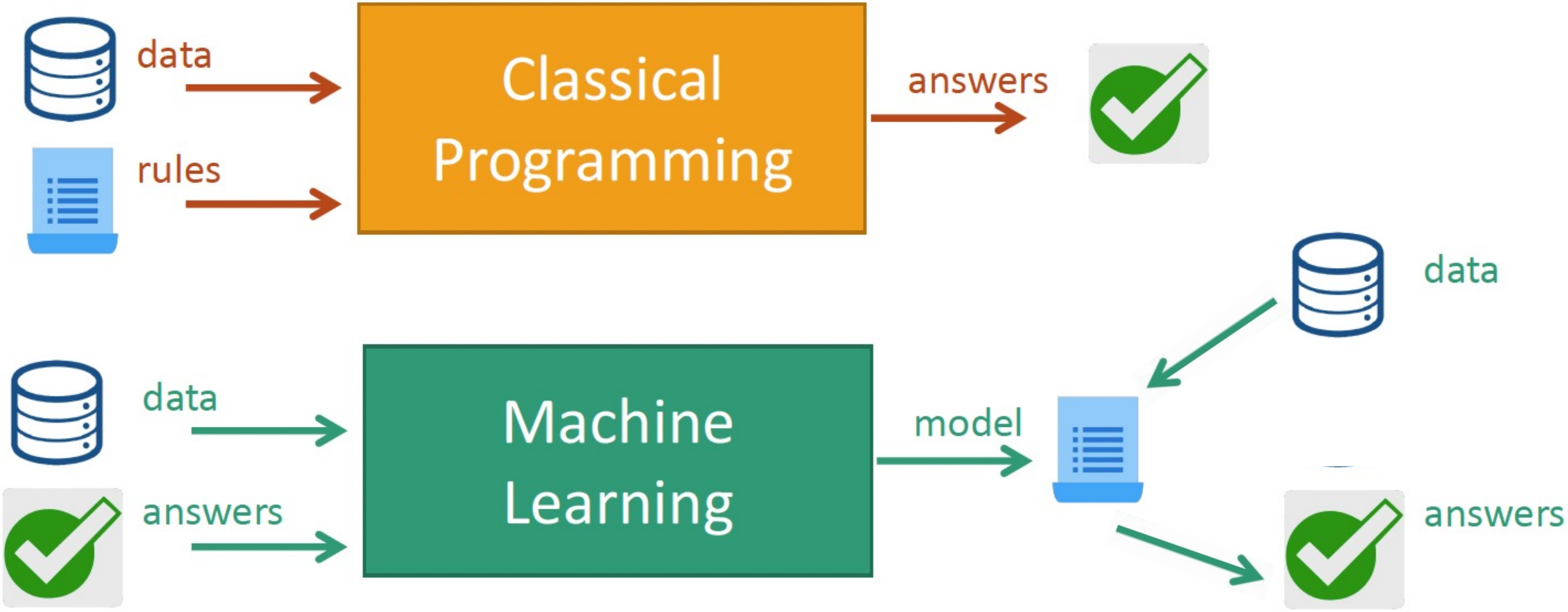
Classical Programming vs Machine Learning

- A ML system is trained rather than programmed

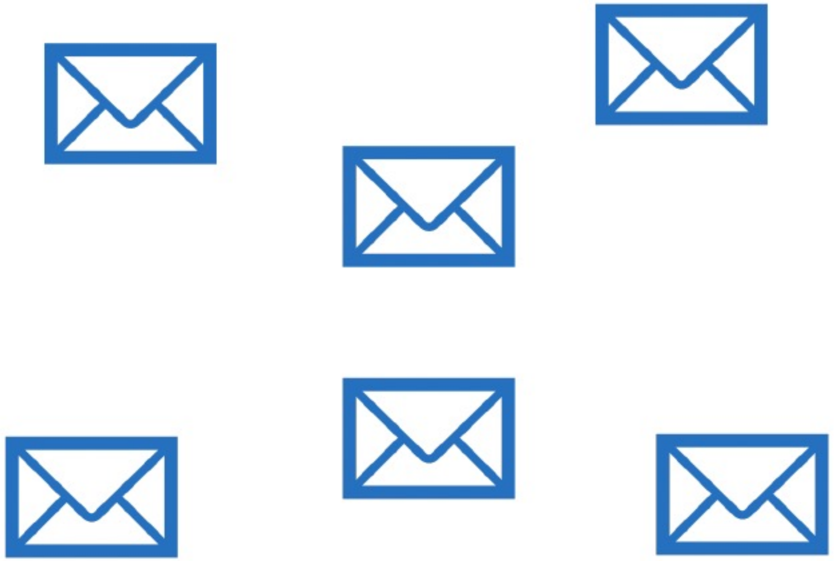


Classical Programming vs Machine Learning

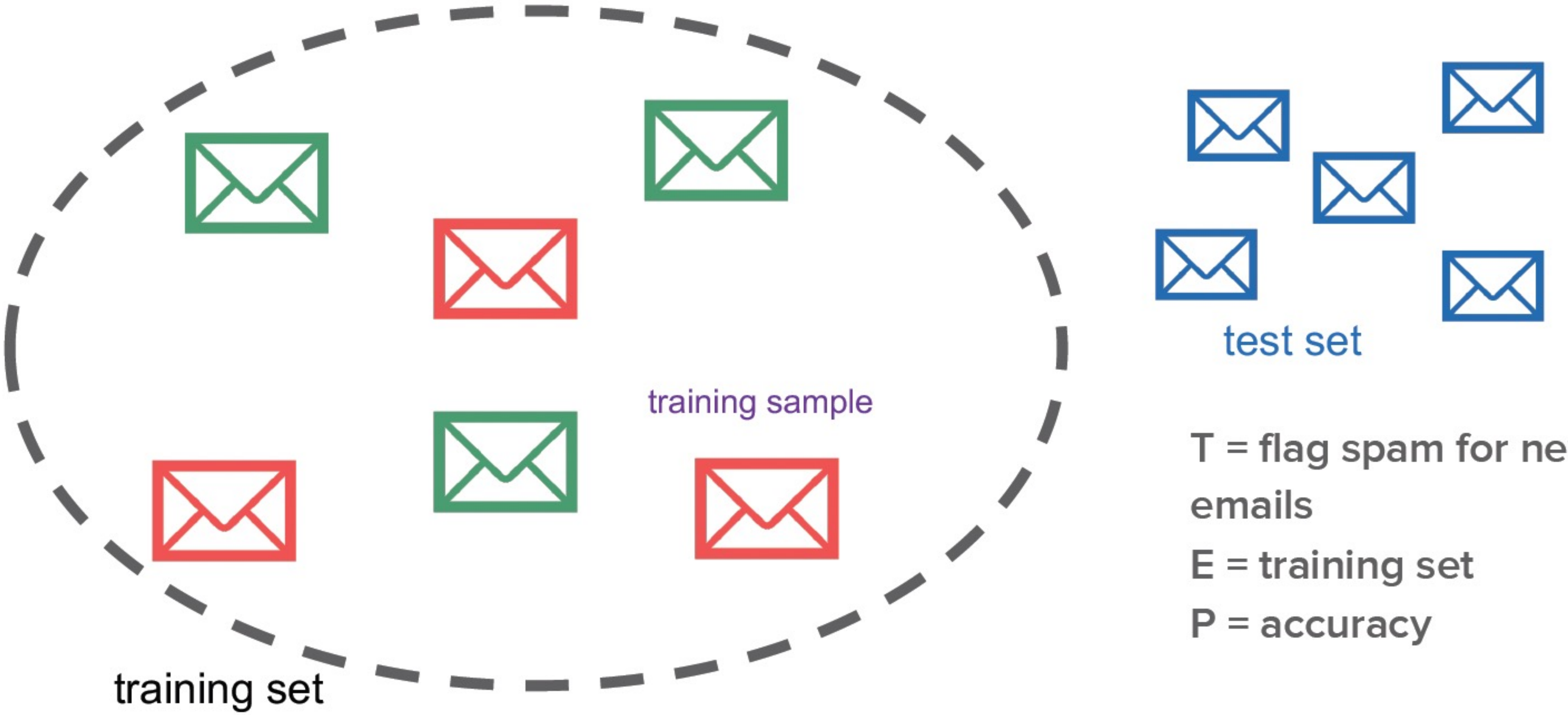
- A ML system is trained rather than programmed



Example Spam Filter



Example Spam Filter

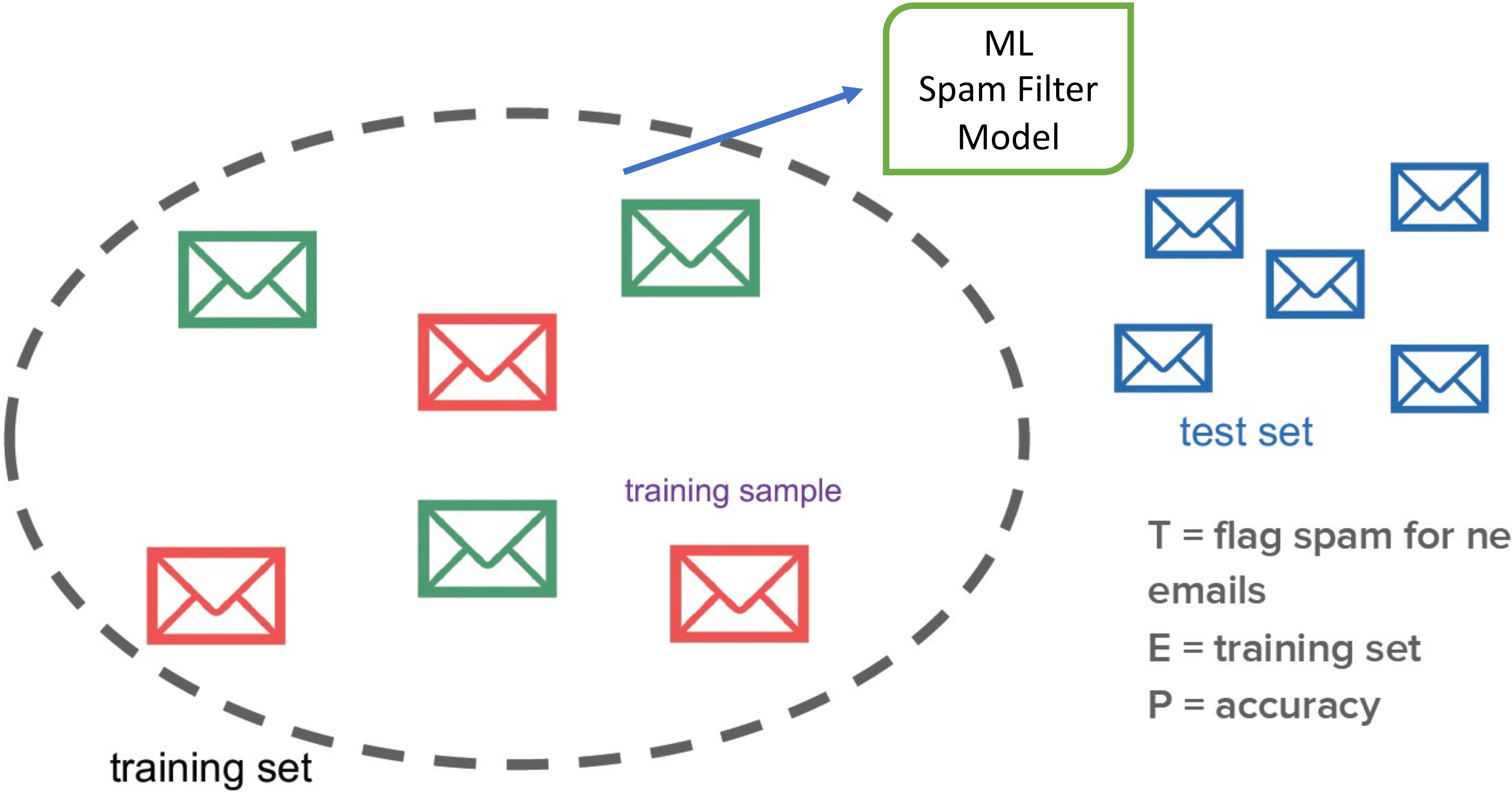


T = flag spam for new emails

E = training set

P = accuracy

Example Spam Filter

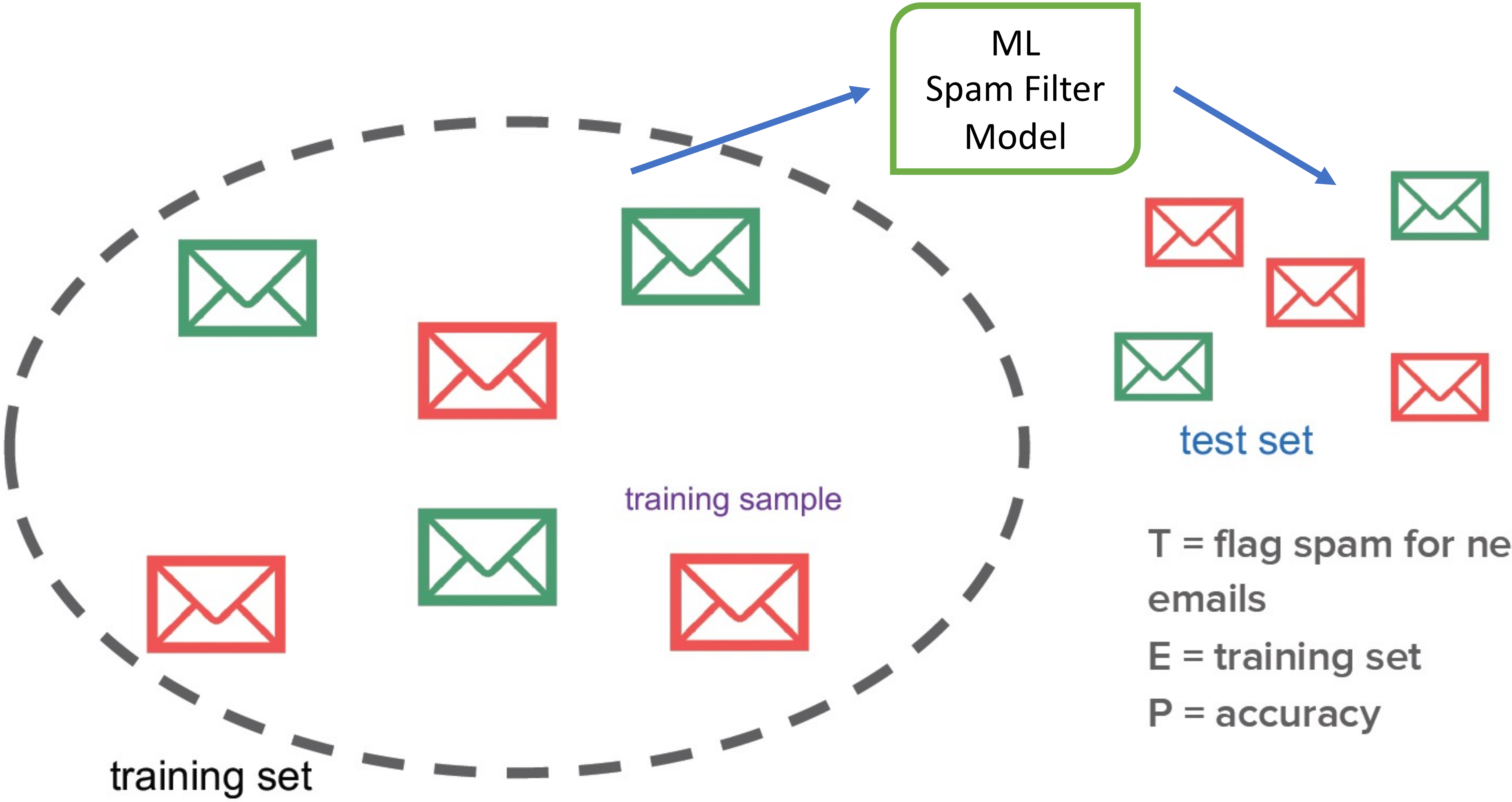


T = flag spam for new emails

E = training set

P = accuracy

Example Spam Filter



ML
Spam Filter
Model

training sample

training set

test set

T = flag spam for new emails

E = training set

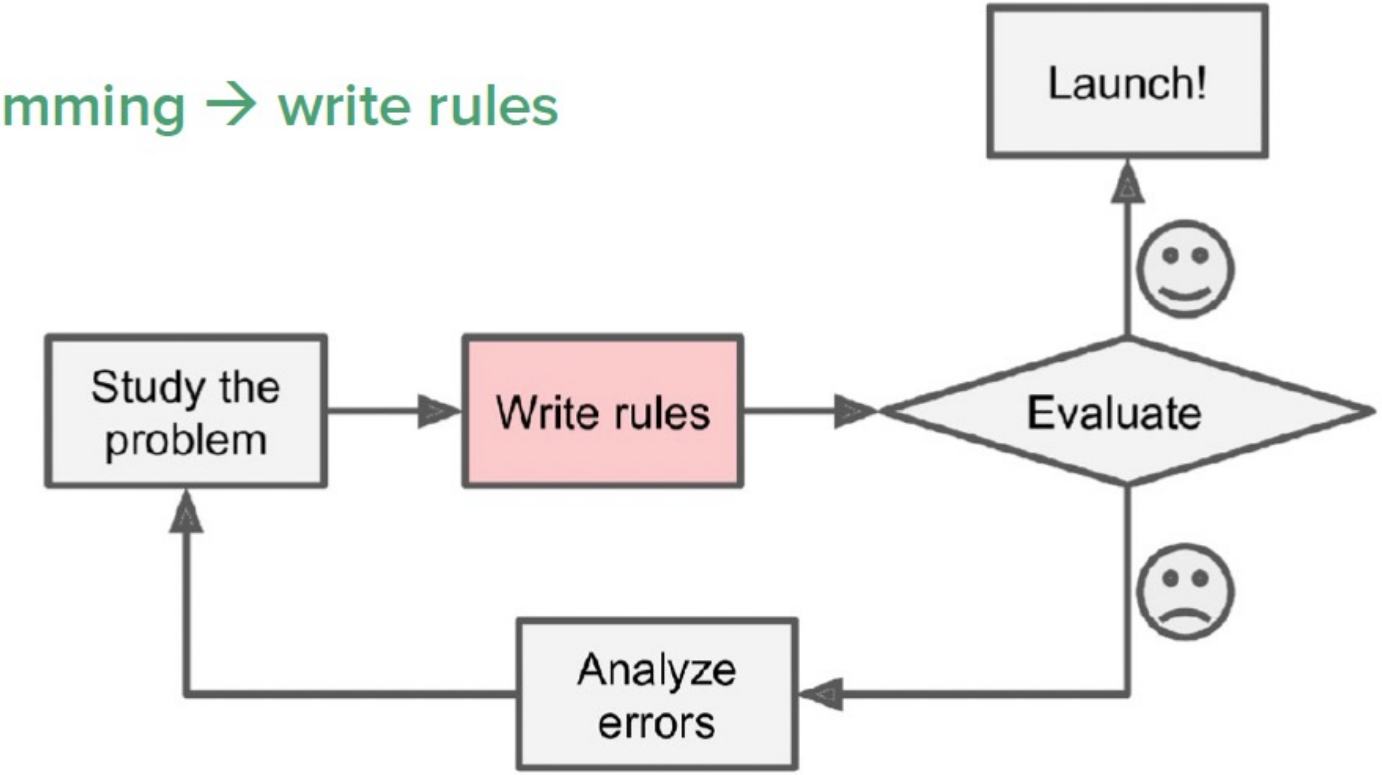
P = accuracy

Examples of Classification Task

Task	Attribute set, $\boldsymbol{x}$	Class label, y
Categorizing email messages	Features extracted from email message header and content	spam or non-spam
Identifying tumor cells	Features extracted from MRI scans	malignant or benign cells
Cataloging galaxies	Features extracted from telescope images	Elliptical, spiral, or irregular-shaped galaxies

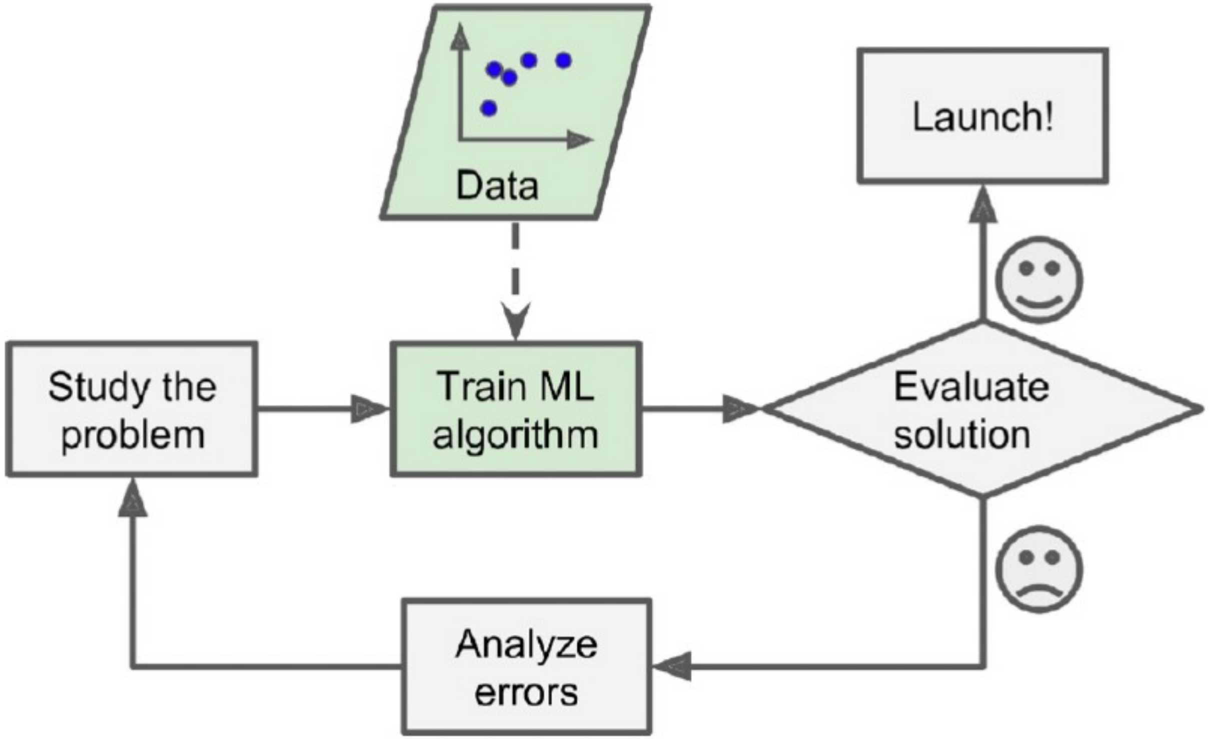
Why do we want to use Machine Learning?

Traditional Programming → write rules



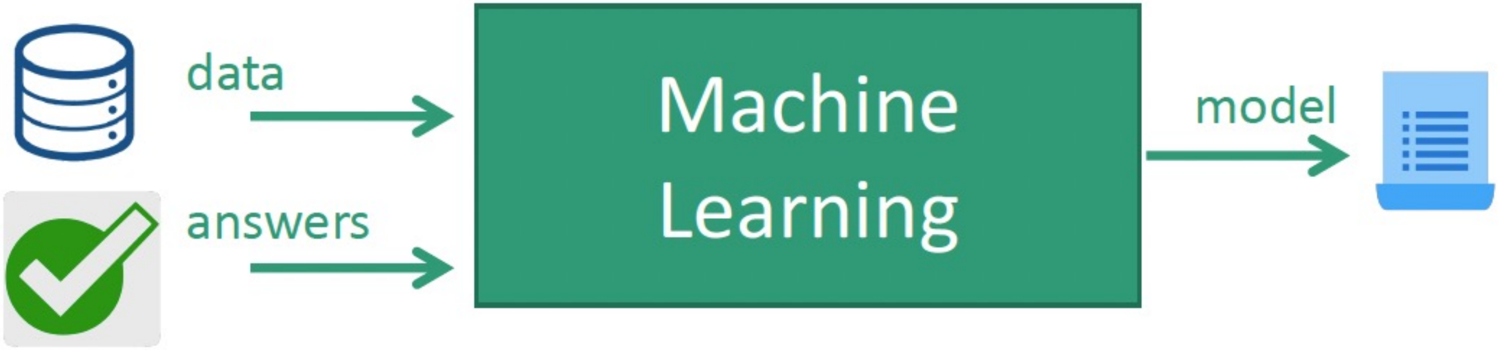
Why do we want to use Machine Learning?

Machine Learning: train based on data (examples)



Why do we want to use Machine Learning?

- Problems for which existing solutions require a lot of finetuning or a long list of rules
- Complex problems for which a traditional approach yields no good solution
- Changing environments
- Getting insights about complex problems and large amount of data



What is Classification?

- Classification consists in learning a **model/function *f*** that maps each attribute set ***x*** into one of the predefined class labels ***y***: *f(x) = y*

What is Classification?

- Classification consists in learning a **model/function *f*** that maps each attribute set ***x*** into one of the predefined class labels ***y****: f(x) = y*
- The ***learning*** is performed on a given a collection of records named ***training set*** where each record is by characterized by a tuple ***(x, y)****,* where ***x*** is the attribute set and ***y*** is the class label
  - x: attribute, predictor, independent variable, input
  - y: class, response, dependent variable, output

What is Classification?

- Classification consists in learning a **model/function *f*** that maps each attribute set ***x*** into one of the predefined class labels ***y****: f(x) = y*
- The ***learning*** is performed on a given a collection of records named ***training set*** where each record is by characterized by a tuple ***(x, y)***, where ***x*** is the attribute set and ***y*** is the class label
    - x: attribute, predictor, independent variable, input
    - y: class, response, dependent variable, output
- **Goal**: previously unseen records should be assigned a class as accurately as possible. A ***test set*** is used to determine the accuracy of the model ***f***.

What is Classification?

- Classification consists in learning a **model/function *f*** that maps each attribute set ***x*** into one of the predefined class labels ***y****: f(x) = y*
- The ***learning*** is performed on a given a collection of records named ***training set*** where each record is by characterized by a tuple ***(x, y)***, where ***x*** is the attribute set and ***y*** is the class label
  - x: attribute, predictor, independent variable, input
  - y: class, response, dependent variable, output
- **Goal**: previously unseen records should be assigned a class as accurately as possible. A ***test set*** is used to determine the accuracy of the model ***f***.
- Usually, the given data set is divided into training and test sets, with training set used to build the model and test set used to validate it.

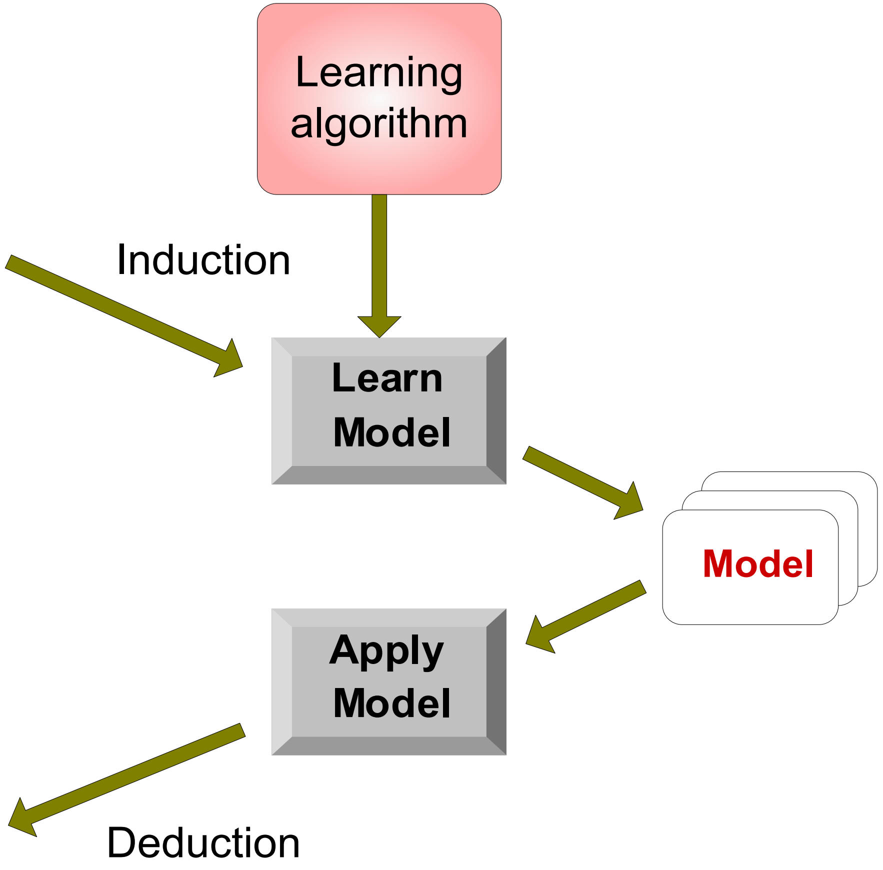
General Approach for Building Classification Model

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	**No**
2	No	Medium	100K	**No**
3	No	Small	70K	**No**
4	Yes	Medium	120K	**No**
5	No	Large	95K	**Yes**
6	No	Medium	60K	**No**
7	Yes	Large	220K	**No**
8	No	Small	85K	**Yes**
9	No	Medium	75K	**No**
10	No	Small	90K	**Yes**

Training Set

Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	**?**
12	Yes	Medium	80K	**?**
13	Yes	Large	110K	**?**
14	No	Small	95K	**?**
15	No	Large	67K	**?**

Test Set



Learning algorithm

Induction

Learn Model

Model

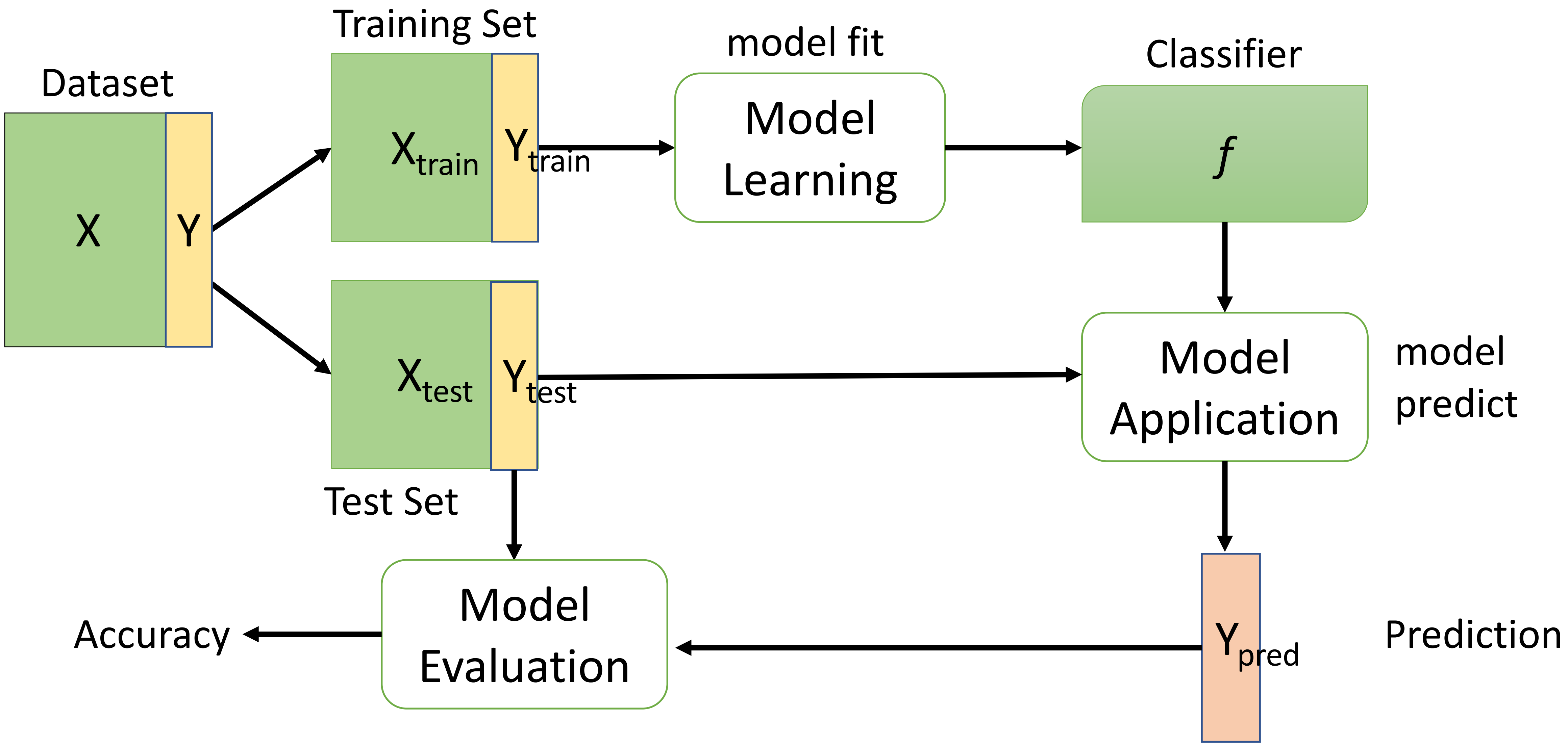
Apply Model

Deduction

Classification Techniques

- Base Classifiers
  - Decision Tree based Methods
  - Rule-based Methods
  - Nearest-neighbor
  - Neural Networks
  - Deep Learning
  - Naïve Bayes and Bayesian Belief Networks
  - Support Vector Machines
- Ensemble Classifiers
  - Boosting, Bagging, Random Forests

What is Classification?



Model Evaluation

Model Evaluation

- Metrics for Performance Evaluation
  - How to evaluate the performance of a model?

- Methods for Performance Evaluation
  - How to obtain reliable estimates?

- Methods for Model Comparison
  - How to compare the relative performance among competing models?

Model Evaluation

- Metrics for Performance Evaluation
  - How to evaluate the performance of a model?
- Methods for Performance Evaluation
  - How to obtain reliable estimates?
- Methods for Model Comparison
  - How to compare the relative performance among competing models?

Problem Setting

- Let suppose we have a vector y of actual/real class labels, i.e.,
- y = [0 0 0 1 1 1 0 1 0 1 0 1 1 1 0 0]

- Let name y' the vector returned by a trained model f, i.e.,
- y' = [0 0 1 1 1 0 0 1 0 1 1 1 0 0 0 0]

Metrics for Performance Evaluation

- Focus on the predictive capability of a model
  - Rather than how fast it takes to classify or build models, scalability, etc.
- **Confusion Matrix:**

	PREDICTED CLASS		
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a	b
	Class=No	c	d

a: TP (true positive)

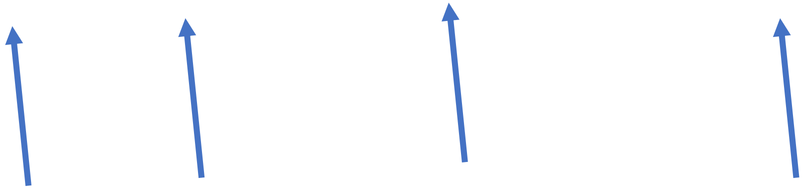
b: FN (false negative)

c: FP (false positive)

d: TN (true negative)

Metrics for Performance Evaluation

- y = [0 0 0 1 1 1 0 1 0 1 0 1 1 1 0 0]
- y' = [0 0 1 1 1 0 0 1 0 1 1 1 0 0 0 0]



- TN FP FN TP

Metrics for Performance Evaluation…

	PREDICTED CLASS		
ACTUAL CLASS		Class=Yes	Class=No
	Class=Yes	a (TP)	b (FN)
	Class=No	c (FP)	d (TN)

Most widely-used metric:

$$\text{Accuracy} = \frac{a+d}{a+b+c+d} = \frac{TP+TN}{TP+TN+FP+FN}$$

Limitation of Accuracy

- Consider a 2-class problem
  - Number of Class 0 examples = 9990
  - Number of Class 1 examples = 10

- If model predicts everything to be class 0, accuracy is 9990/10000 = 99.9 %
- Accuracy is misleading because model does not detect any class 1 example

Cost-Sensitive Measures

$$\text{Precision (p)} = \frac{TP}{TP + FP}$$

$$\text{Recall (r)} = \frac{TP}{TP + FN}$$

$$\text{F-measure (F)} = \frac{2rp}{r + p} = \frac{2TP}{2TP + FN + FP}$$

- Precision is biased towards C(Yes|Yes) & C(Yes|No)
- Recall is biased towards C(Yes|Yes) & C(No|Yes)
- F-measure is biased towards all except C(No|No)

$$\text{Weighted Accuracy} = \frac{w_1 a + w_4 d}{w_1 a + w_2 b + w_3 c + w_4 d}$$

Cost Matrix

<table>
<tr><td></td><td colspan="3">PREDICTED CLASS</td></tr>
<tr><td rowspan="3">ACTUAL CLASS</td><td>C(i|j)</td><td><b>Class=Yes</b></td><td><b>Class=No</b></td></tr>
<tr><td><b>Class=Yes</b></td><td>C(Yes|Yes)</td><td>C(No|Yes)</td></tr>
<tr><td><b>Class=No</b></td><td>C(Yes|No)</td><td>C(No|No)</td></tr>
</table>

C(i|j): Cost of misclassifying class j example as class i

Computing Cost of Classification

Cost Matrix	PREDICTED CLASS		
ACTUAL CLASS	C(i|j)	+	-
	+	-1	100
	-	1	0

Model M_1	PREDICTED CLASS		
ACTUAL CLASS		+	-
	+	150	40
	-	60	250

Model M_2	PREDICTED CLASS		
ACTUAL CLASS		+	-
	+	250	45
	-	5	200

Accuracy = 80%

Cost = 3910

Accuracy = 90%

Cost = 4255

Cost vs Accuracy

Count		PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a	b
	Class=No	c	d

Cost		PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	p	q
	Class=No	q	p

Accuracy is proportional to cost if
1. C(Yes|No)=C(No|Yes) = q
2. C(Yes|Yes)=C(No|No) = p

$N = a + b + c + d$

$\text{Accuracy} = (a + d)/N$

$$
\begin{aligned}
\text{Cost} &= p\,(a + d) + q\,(b + c) \\
&= p\,(a + d) + q\,(N - a - d) \\
&= q\,N - (q - p)(a + d) \\
&= N\,[q - (q-p) \times \text{Accuracy}]
\end{aligned}
$$

Binary vs Multiclass Evaluation

	PREDICTED CLASS		
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	TP	FN
	Class=No	FP	TN

	PREDICTED CLASS			
		Class=A	Class=B	Class=C
ACTUAL CLASS	Class=A	TP-A		
	Class=B		TP-B	
	Class=C			TP-C

Accuracy = TP+TN / (TP+TN+FN+FP) = # correct / N

Accuracy = # correct / N = (TP-A + TP-B + TP-C) / N

Multiclass Evaluation

		PREDICTED CLASS		
ACTUAL CLASS		Class=A	Class=B	Class=C
	Class=A	TP-A	a	b
	Class=B	c	TP-B	d
	Class=C	e	f	TP-C

$$\text{Precision (p)} = \frac{TP}{TP + FP}$$

$$\text{Recall (r)} = \frac{TP}{TP + FN}$$

$$\text{F-measure (F)} = \frac{2rp}{r + p} = \frac{2TP}{2TP + FN + FP}$$

A		PREDICTED CLASS	
ACTUAL CLASS		Class=A	Class=Not A
	Class=A	TP-A	a + b
	Class=Not A	c + e	TP-B + TP-C + d + f

B		PREDICTED CLASS	
ACTUAL CLASS		Class=B	Class=Not B
	Class=B	TP-B	c + d
	Class=Not B	a + f	TP-A + TP-C + b + e

C		PREDICTED CLASS	
ACTUAL CLASS		Class=C	Class=Not C
	Class=C	TP-C	e + f
	Class=Not C	b + d	TP-A + TP-B + a + c

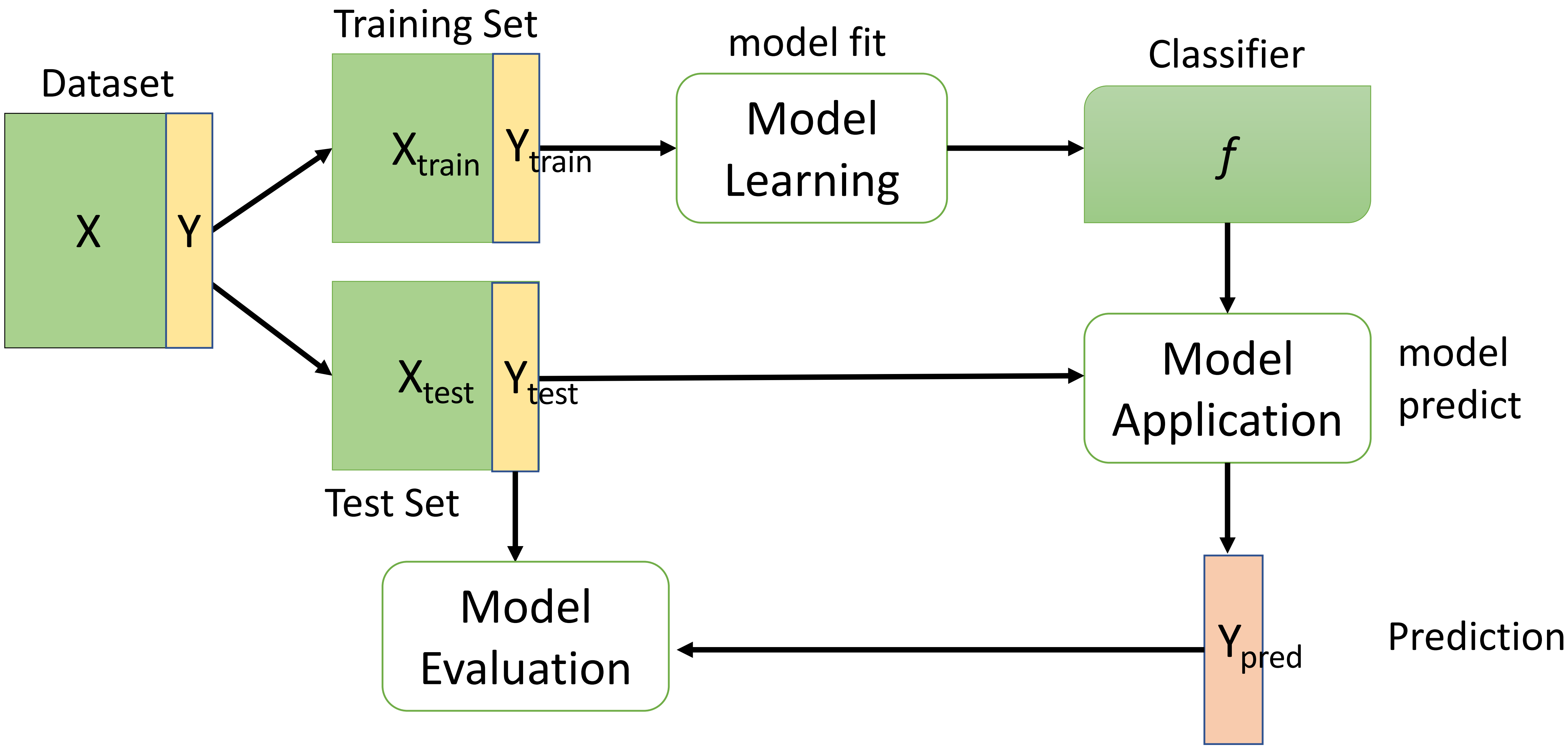
Model Evaluation

- Metrics for Performance Evaluation
  - How to evaluate the performance of a model?

- Methods for Performance Evaluation
  - How to obtain reliable estimates?

- Methods for Model Comparison
  - How to compare the relative performance among competing models?

Methods for Evaluation



Parameter Tuning

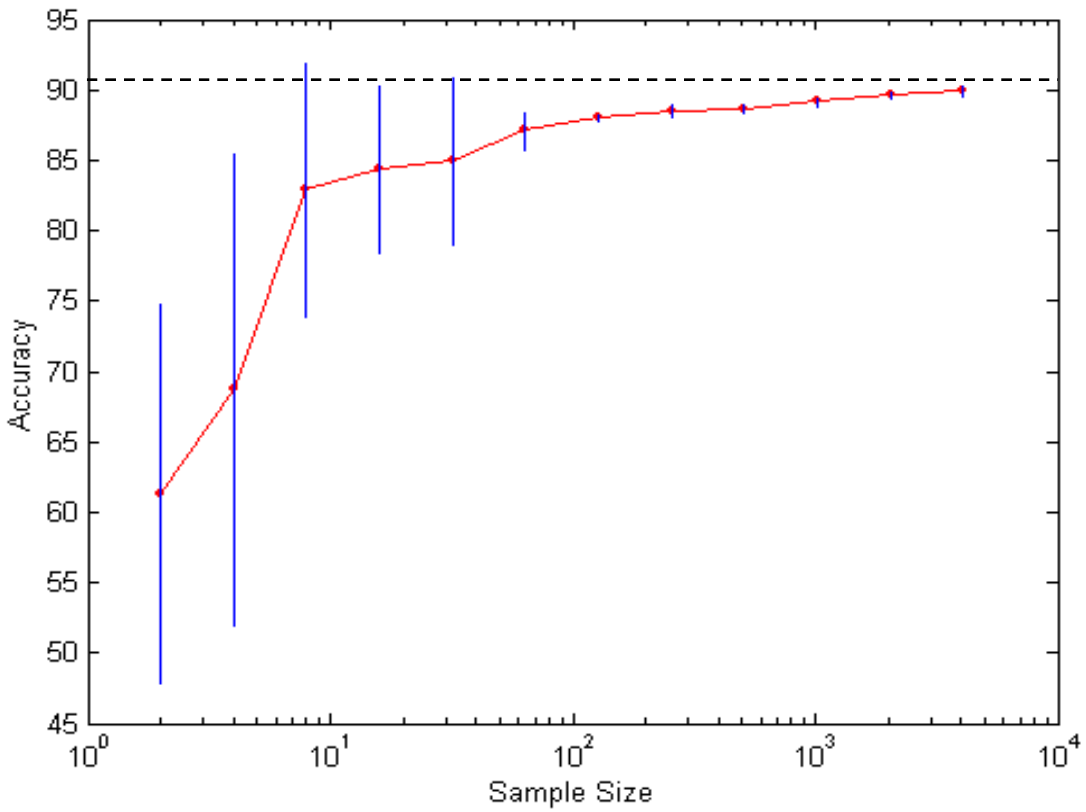
- It is important that the test data is not used in any way to create the classifier
- Some learning schemes operate in two stages:
  - **Stage 1**: builds the basic structure
  - **Stage 2**: optimizes parameter settings
  - **The test data can’t be used for parameter tuning!**
  - Proper procedure uses three sets:
    - training data,
    - validation data,
    - test data
  - **Validation data is used to optimize parameters**
- Once evaluation is complete, all the data can be used to build the final classifier
- Generally, the larger the training data the better the classifier
- The larger the test data the more accurate the error estimate

Methods for Performance Evaluation

- How to obtain a reliable estimate of performance?

- Performance of a model may depend on other factors besides the learning algorithm:
  - Class distribution
  - Cost of misclassification
  - Size of training and test sets

Learning Curve



- Learning curve shows how accuracy changes with varying sample size
- Requires a sampling schedule for creating learning curve:

Effect of small sample size:

- Bias in the estimate
- Variance of estimate

1. How much a classification model benefits from adding more training data?
2. Does the model suffer from a variance error or a bias error?

Methods of Estimation

- Holdout
  - Reserve 2/3 for training and 1/3 for testing
- Random subsampling
  - Repeated holdout
- Cross validation
  - Partition data into k disjoint subsets
  - k-fold: train on k-1 partitions, test on the remaining one
  - Leave-one-out: k=n
- Stratified sampling
  - oversampling vs undersampling
- Bootstrap
  - Sampling with replacement

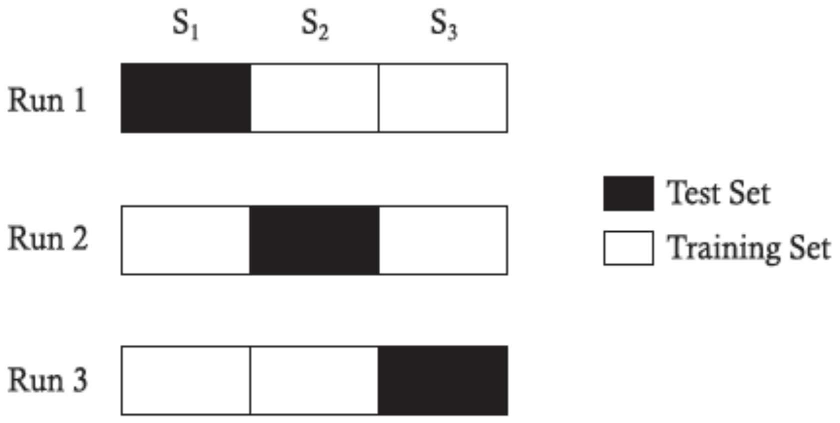
Small & Unbalanced Data

- The holdout method reserves a certain amount for **testing** and uses the remainder for **training**
- Usually, **one third for testing**, the rest for training
- For small or “unbalanced” datasets, **samples might not be representative**
    - For instance, few or none instances of some classes
- Stratified sample
    - **Balancing the data**
    - Make sure that each class is represented with approximately equal proportions in both subsets

Repeated holdout method

- Holdout estimate can be made more reliable by **repeating the process with different subsamples**
  - In each iteration, a certain proportion is **randomly selected for training** (possibly with stratification)
  - The error rates on the different iterations are **averaged** to yield an overall error rate
- This is called the **repeated holdout method**
- Still not optimum: the different test sets overlap

Cross Validation



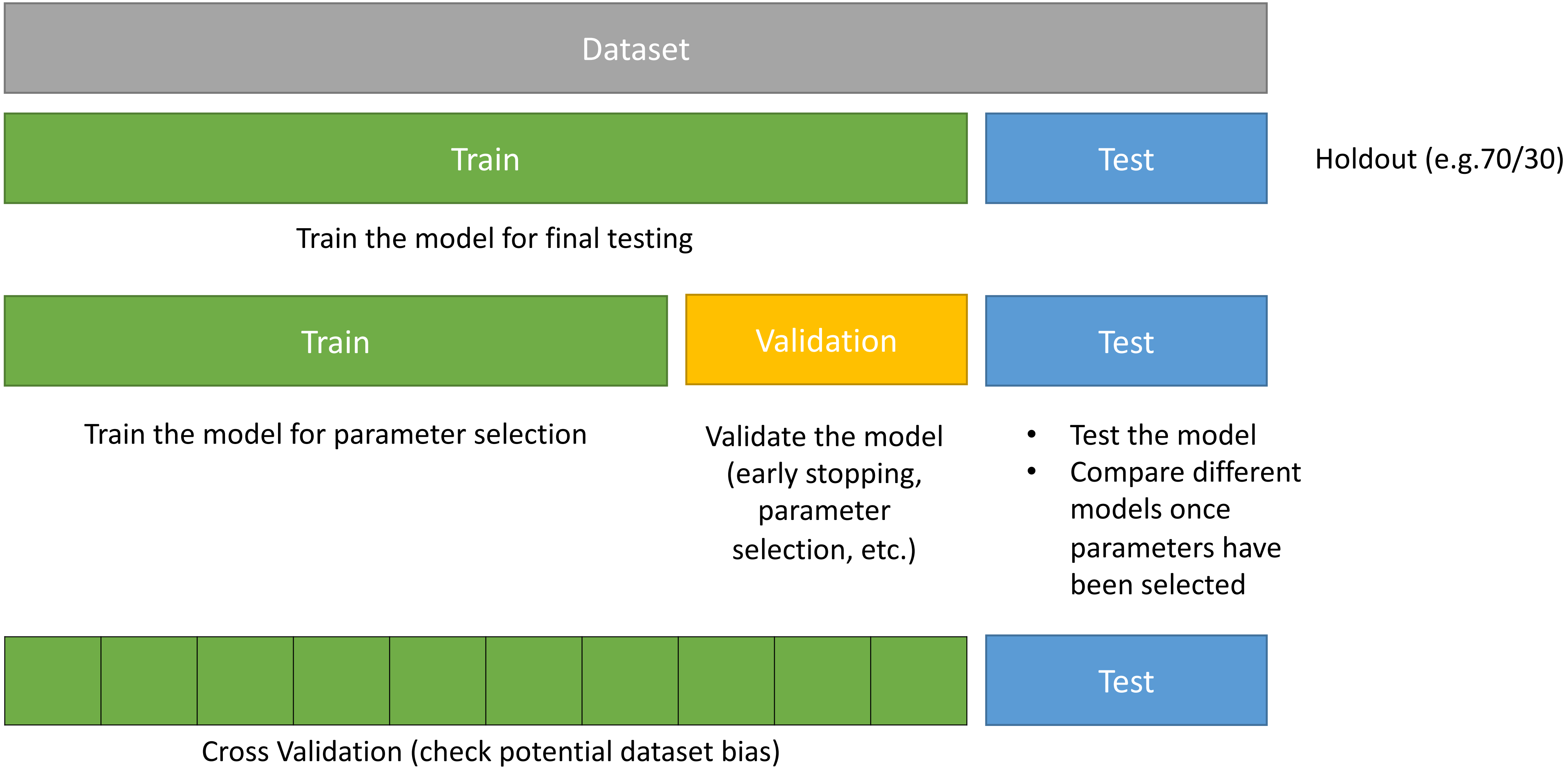
- Avoids overlapping test sets
  - **First step:** data is split into k subsets of equal size
  - **Second step:** each subset in turn is used for testing and the remainder for training
- This is called **k-fold cross-validation**

- Often the subsets are stratified before cross-validation is performed

- The **error estimates** are **averaged** to yield an overall error estimate

- **Even better:** repeated stratified cross-validation E.g. ten-fold cross-validation is repeated ten times and results are averaged (reduces the variance)

Data Partitioning



Dataset

Train | Test — Holdout (e.g.70/30)

Train the model for final testing

Train | Validation | Test

Train the model for parameter selection

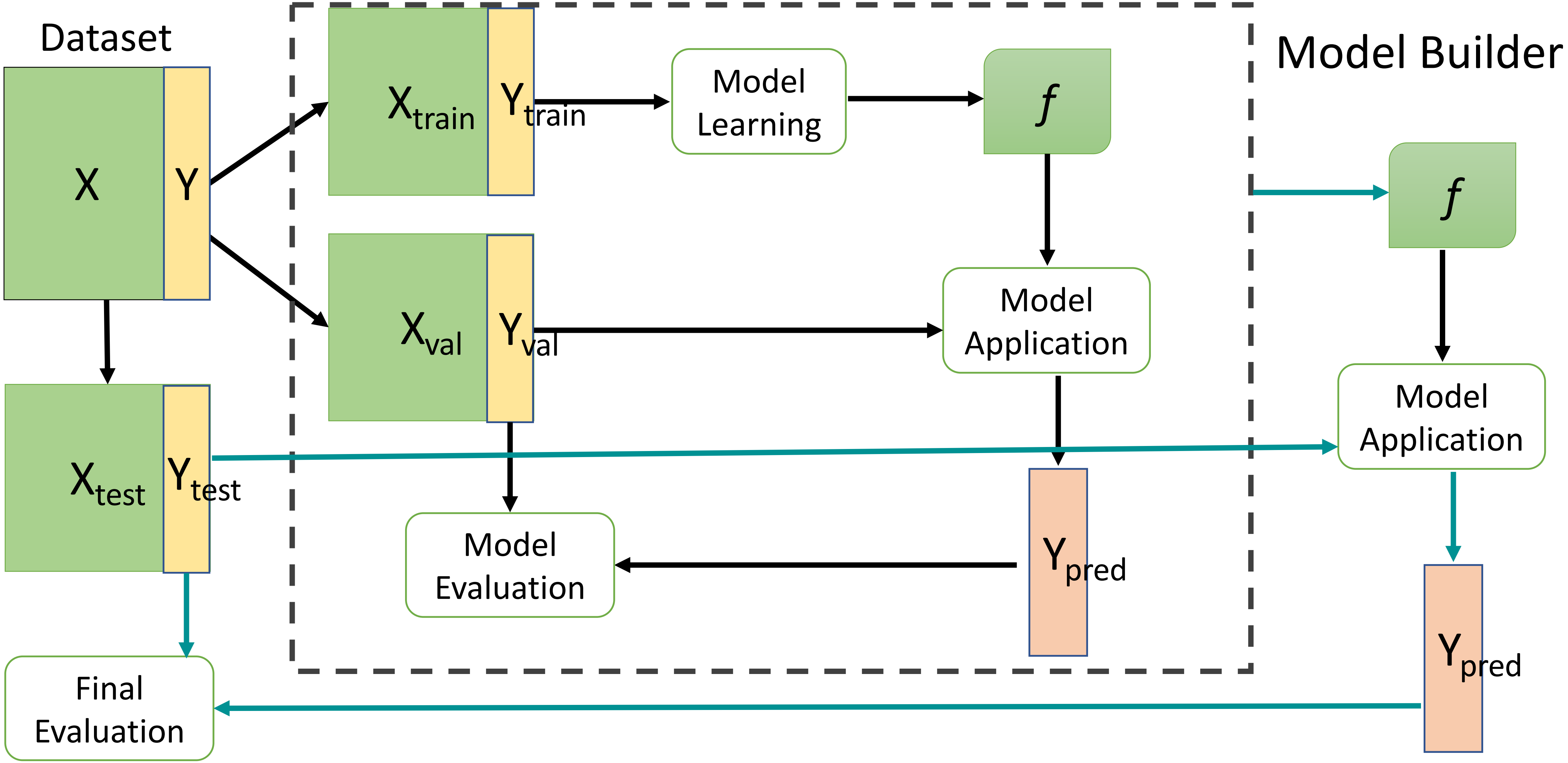
Validate the model (early stopping, parameter selection, etc.)

- Test the model
- Compare different models once parameters have been selected

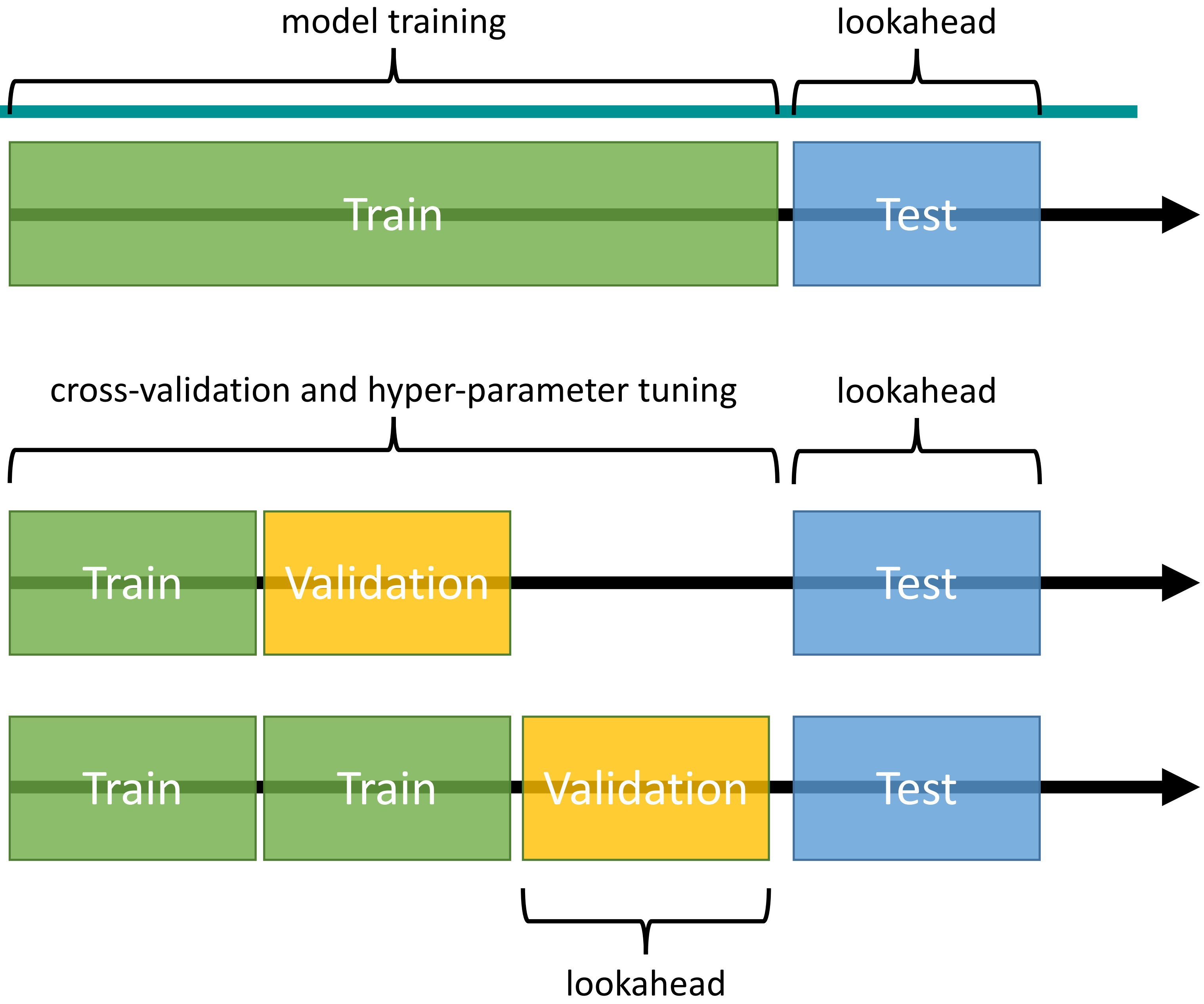
Test

Cross Validation (check potential dataset bias)

Evaluation: Training, Validation, Tests



Cross Validation with Time



Model Evaluation

- Metrics for Performance Evaluation
  - How to evaluate the performance of a model?
- Methods for Performance Evaluation
  - How to obtain reliable estimates?
- Methods for Model Comparison
  - How to compare the relative performance among competing models?

ROC (Receiver Operating Characteristic)

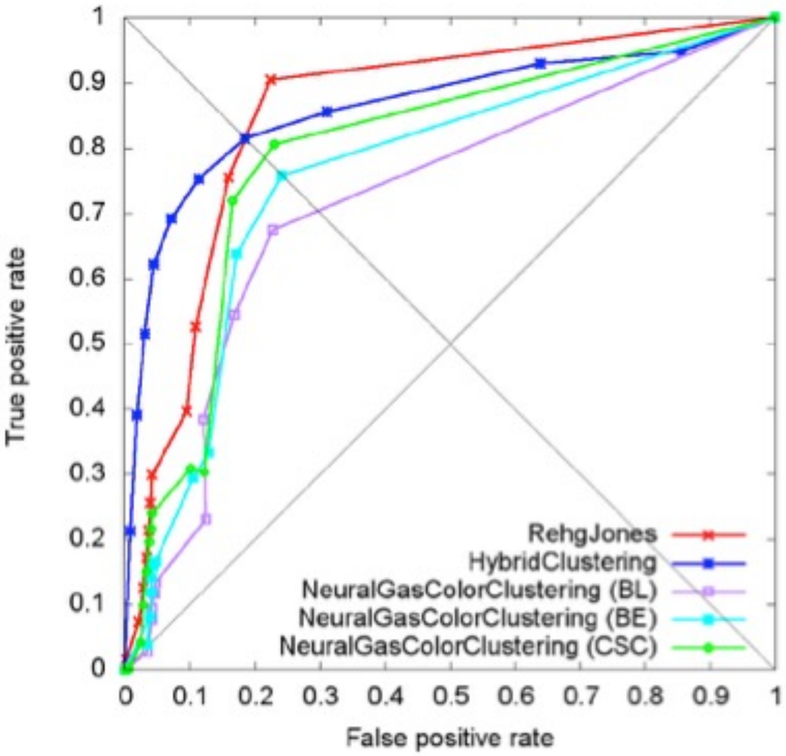
- Developed in 1950s for signal detection theory to analyze noisy signals
  - Characterize the trade-off between positive hits and false alarms

- ROC curve plots TP (on the y-axis) against FP (on the x-axis)

- **Performance of each classifier represented as a point on the ROC curve**
  - changing the threshold of algorithm, sample distribution or cost matrix changes the location of the point

Receiver Operating Characteristic Curve

- It illustrates the ability of a binary classifier as its discrimination threshold THR is varied.
- The ***ROC*** curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various THR.
- The TPR = TP / (TP + FN) is also known as ***sensitivity***, ***recall*** or probability of detection.
- The FPR = FP / (TN + FP) is also known as probability of ***false alarm*** and can be calculated as (1 – specificity).



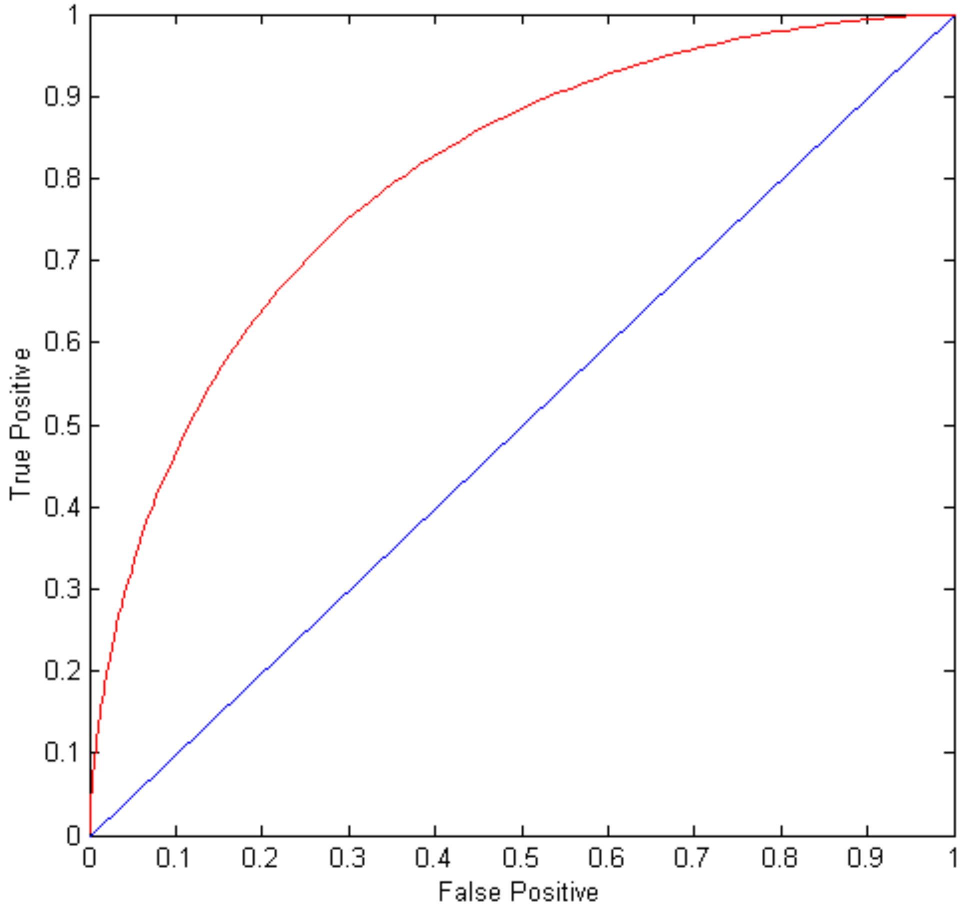
https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5

ROC Curve

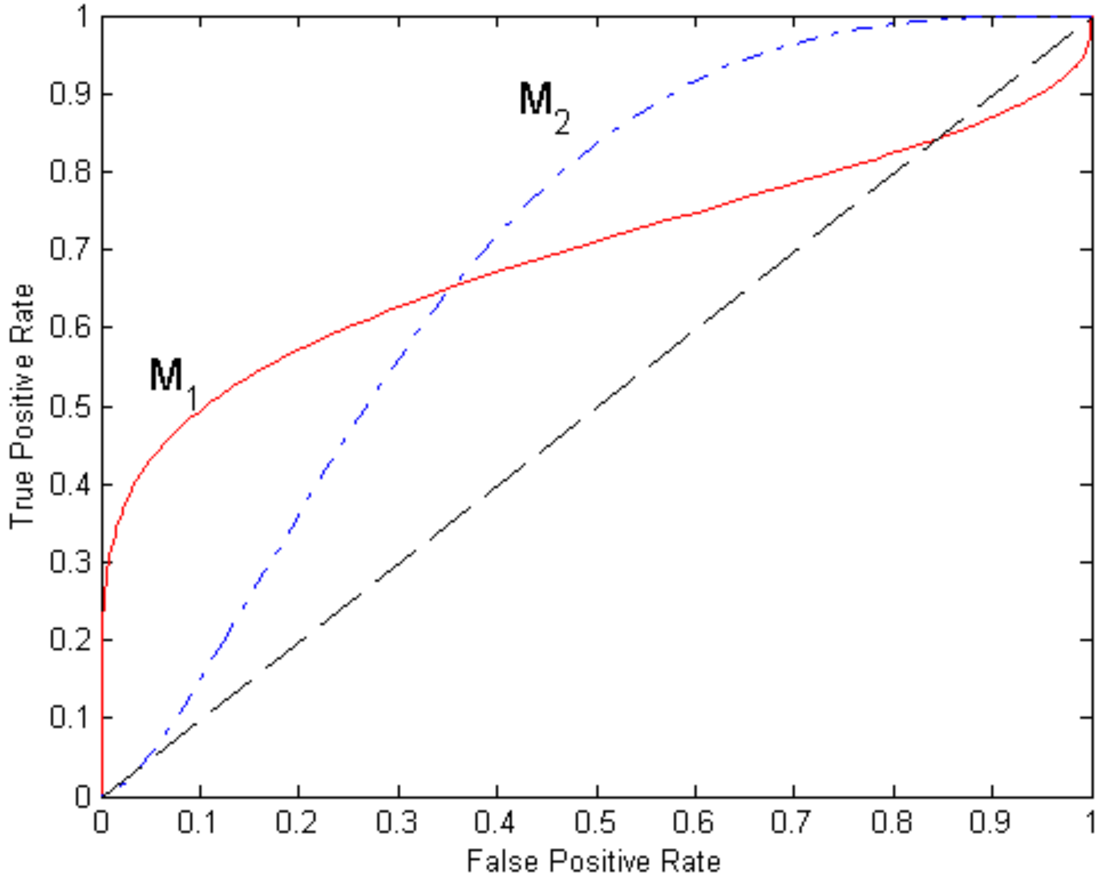
(TP,FP):

- (0,0): declare everything
  to be negative class
- (1,1): declare everything
  to be positive class
- (0,1): ideal

- Diagonal line:
  - Random guessing
  - Below diagonal line:
    - prediction is opposite of the true class



Using ROC for Model Comparison



- No model consistently outperform the other
  - M$_1$ is better for small FPR
  - M$_2$ is better for large FPR
- Area Under the ROC curve
  - Ideal:
    - Area = 1
  - Random guess:
    - Area = 0.5

How to Construct the ROC curve

Instance	P(+|A)	True Class
1	0.95	+
2	0.93	+
3	0.87	-
4	0.85	-
5	0.85	-
6	0.85	+
7	0.76	-
8	0.53	+
9	0.43	-
10	0.25	+

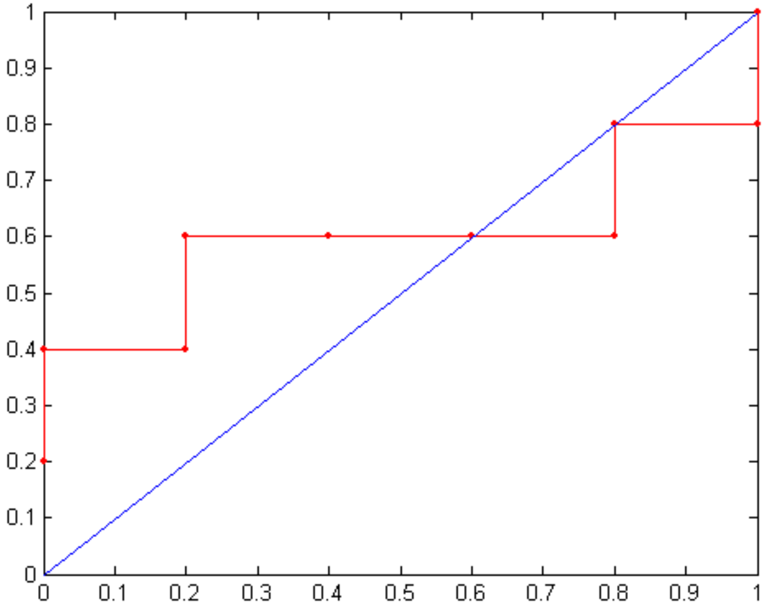
- Use classifier that produces posterior probability for each test instance P(+|A)
- Sort the instances according to P(+|A) in decreasing order
- Apply threshold at each unique value of P(+|A)
- Count the number of TP, FP, TN, FN at each threshold
- TP rate, TPR = TP/(TP+FN)
- FP rate, FPR = FP/(FP + TN)

How to Construct the ROC curve

Class	+	-	+	-	-	-	+	-	+	+	
Threshold >=	0.25	0.43	0.53	0.76	0.85	0.85	0.85	0.87	0.93	0.95	1.00
TP	5	4	4	3	3	3	3	2	2	1	0
FP	5	5	4	4	3	2	1	1	0	0	0
TN	0	0	1	1	2	3	4	4	5	5	5
FN	0	1	1	2	2	2	2	3	3	4	5
TPR	1	0.8	0.8	0.6	0.6	0.6	0.6	0.4	0.4	0.2	0
FPR	1	1	0.8	0.8	0.6	0.4	0.2	0.2	0	0	0

TPR = TP / (TP + FN)
FPR = FP / (TN + FP)

ROC Curve:



Inst.	P(+|A)	True Class
1	0.95	+
2	0.93	+
3	0.87	-
4	0.85	-
5	0.85	-
6	0.85	+
7	0.76	-
8	0.53	+
9	0.43	-
10	0.25	+

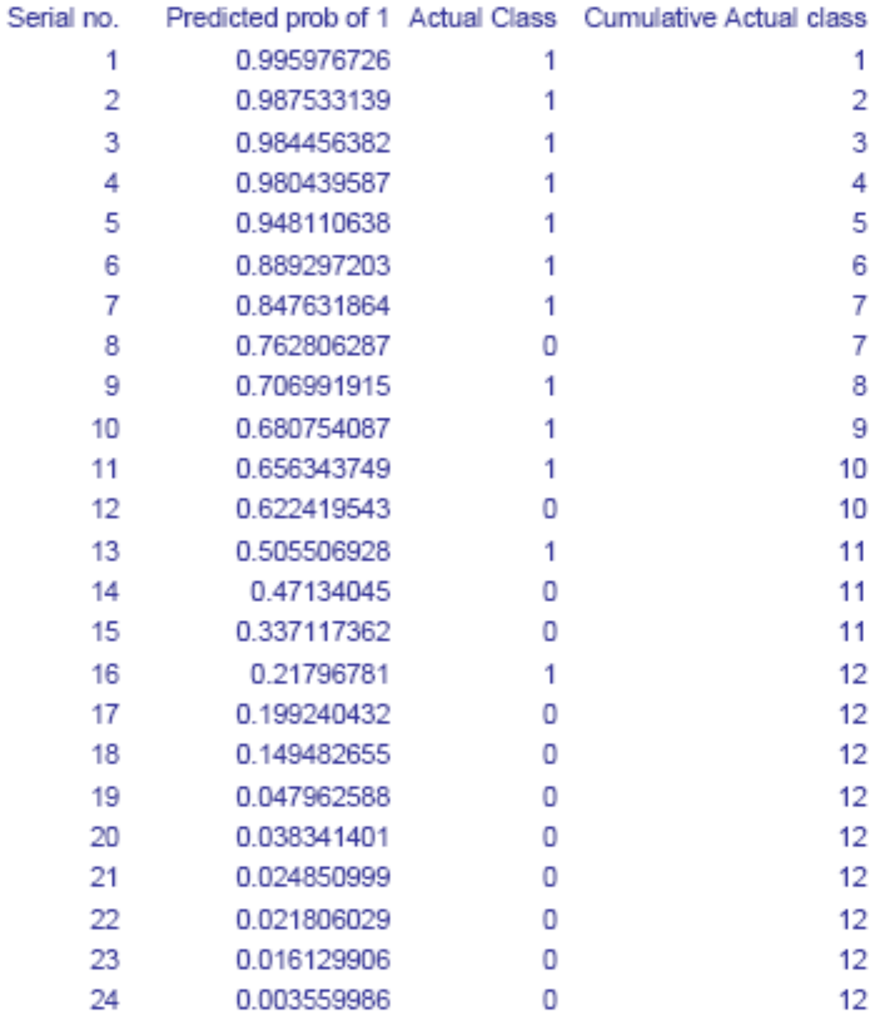
Lift Chart

http://www2.cs.uregina.ca/~dbd/cs831/notes/lift_chart/lift_chart.html
http://mlwiki.org/index.php/Cumulative_Gain_Chart

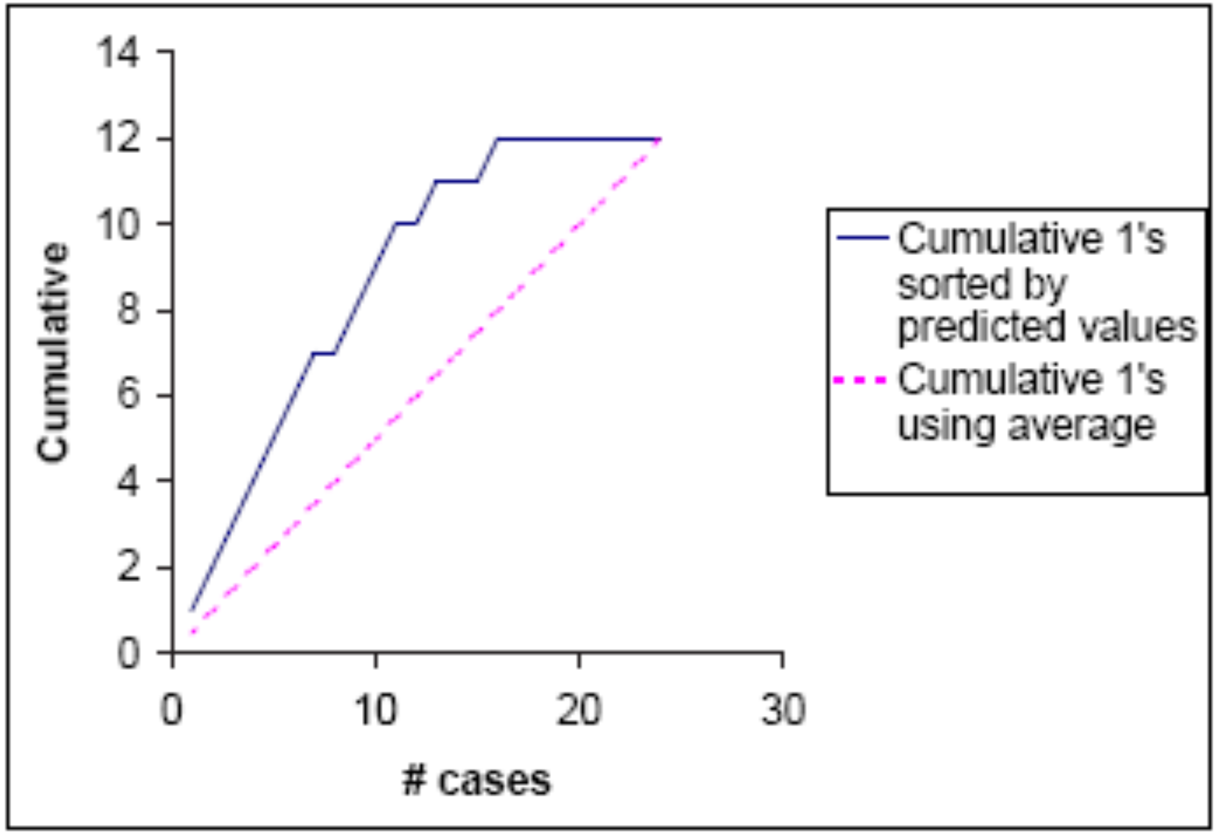
- The lift curve is a popular technique in direct marketing.
- The input is a dataset that has been “scored’’ by appending to each case the estimated probability that it will belong to a given class.
- The cumulative ***lift chart*** (also called ***gains chart***) is constructed with the cumulative number of cases (descending order of probability) on the x-axis and the cumulative number of true positives on the y-axis.
- The dashed line is a reference line. For any given number of cases (the x-axis value), it represents the expected number of positives we would predict if we did not have a model but simply selected cases at random. It provides a benchmark against which we can see performance of the model.

Notice: “Lift chart” is a rather general term, often used to identify also other kinds of plots. Don’t get confused!

Lift Chart – Example



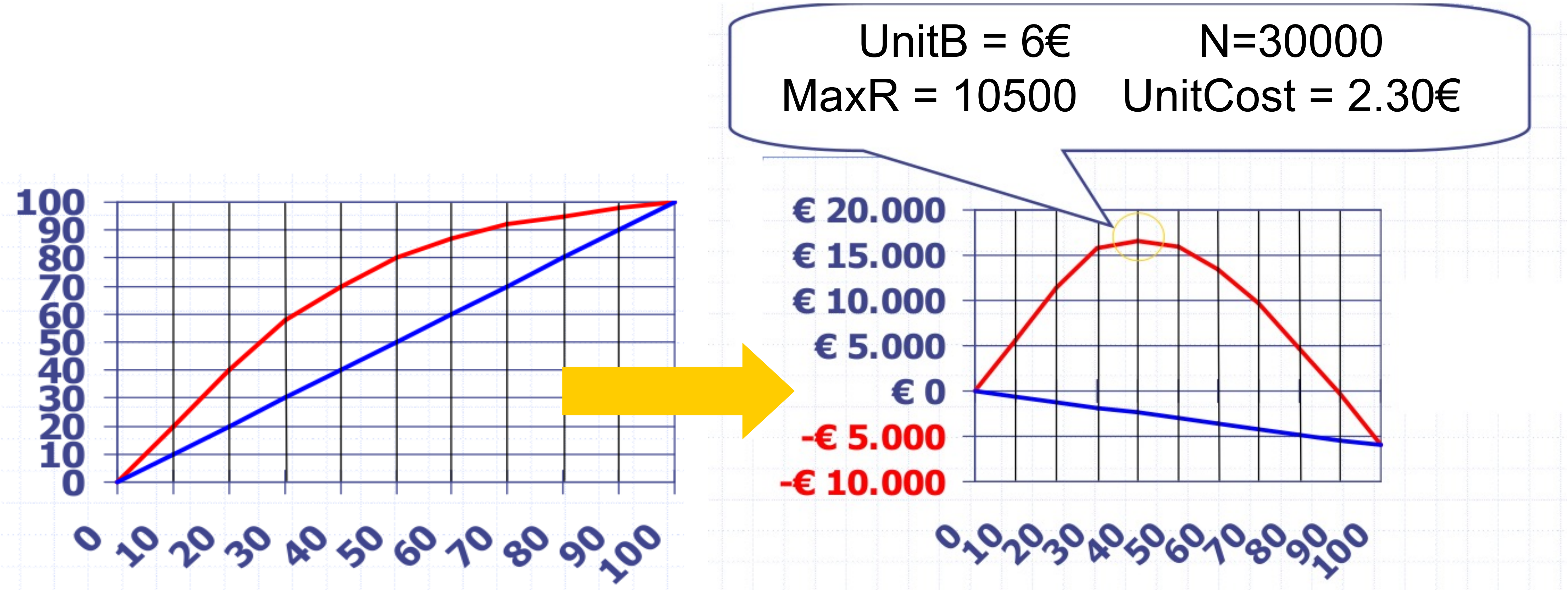
Serial no.	Predicted prob of 1	Actual Class	Cumulative Actual class
1	0.995976726	1	1
2	0.987533139	1	2
3	0.984456382	1	3
4	0.980439587	1	4
5	0.948110638	1	5
6	0.889297203	1	6
7	0.847631864	1	7
8	0.762806287	0	7
9	0.706991915	1	8
10	0.680754087	1	9
11	0.656343749	1	10
12	0.622419543	0	10
13	0.505506928	1	11
14	0.47134045	0	11
15	0.337117362	0	11
16	0.21796781	1	12
17	0.199240432	0	12
18	0.149482655	0	12
19	0.047962588	0	12
20	0.038341401	0	12
21	0.024850999	0	12
22	0.021806029	0	12
23	0.016129906	0	12
24	0.003559986	0	12



Lift Chart – Application Example

- From Lift chart we can easily derive an “economical value” plot, e.g. in target marketing.
- Given our predictive model, how many customers should we target to maximize income?
- Profit = UnitB*MaxR*Lift(X) - UnitCost*N*X/100
- UnitB = unit benefit, UnitCost = unit postal cost
- N = total customers
- MaxR = expected potential respondents in all population (N)
- Lift(X) = lift chart value for X, in [0,..,1]

Lift Chart – Application Example



UnitB = 6€ N=30000

MaxR = 10500 UnitCost = 2.30€

References

- Chapter 3. Classification: Basic Concepts and Techniques.